SQL

Bases de Datos Avanzadas

Universidad del Rosario

Maestría MACC

SQL como Lenguaje DDL y DDM

1. 

2.

3. Como lenguaje DDL:
   - Tipos de valores en atributos.
   - Ligaduras de integridad.

SQL como Lenguaje DDL y DDM

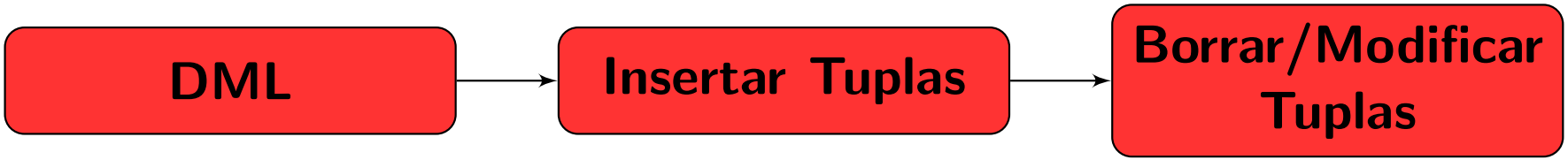
I. 

II. 

III. Como lenguaje DDL:
- Tipos de valores en atributos.
- Ligaduras de integridad.

SQL como Lenguaje DDL y DDM

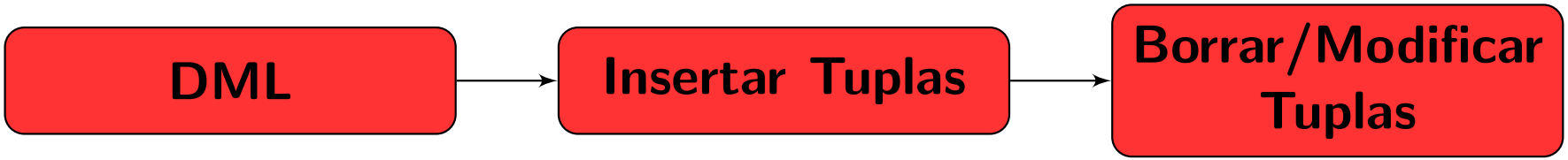
I. 

II. 

III. Como lenguaje DDL:
- Tipos de valores en atributos.
- Ligaduras de integridad.

SQL como Lenguaje DDL y DDM

I. 

II. 

III. Como lenguaje DDL:
- Tipos de valores en atributos.
- Ligaduras de integridad.

Tipos de Datos en SQL

I. char(n) (*n* fijo).
II. varchar(n) (variable -máx *n*-)
III. int/integer, smallint, bigint, tinyint.
IV. float(n), numeric(*p*,*d*).
V. *null*.
   - char/varchar: caracteres.
   - int/integer, smallint, $\cdots$: valores numéricos.

VI. *Esquema de tablas:*

```
create table departamento
(nombre_depto varchar(20),
edificio varchar(25),
presupuesto numeric(12,2),
primary key(nombre_depto));
```
(1)

Tipos de Datos en SQL

I. char(n) (*n* fijo).
II. varchar(n) (variable -máx *n*-)
III. int/integer, smallint, bigint, tinyint.
IV. float(n), numeric(*p*,*d*).
V. *null*.
   - char/varchar: caracteres.
   - int/integer, smallint, $\cdots$: valores numéricos.

VI. *Esquema de tablas:*

```
create table departamento
(nombre_depto varchar(20),
edificio varchar(25),
presupuesto numeric(12,2),
primary key(nombre_depto));
```
(1)

Tipos de Datos en SQL

I. char(n) (*n* fijo).
II. varchar(n) (variable -máx *n*-)
III. int/integer, smallint, bigint, tinyint.
IV. float(n), numeric(*p*,*d*).
V. *null*.
   - char/varchar: caracteres.
   - int/integer, smallint, $\cdots$: valores numéricos.
VI. *Esquema de tablas:*

```
create table departamento
(nombre_depto varchar(20),
edificio varchar(25),
presupuesto numeric(12,2),
primary key(nombre_depto));
```
(1)

Tipos de Datos en SQL

I. char(n) (*n* fijo).
II. varchar(n) (variable -máx *n*-)
III. int/integer, smallint, bigint, tinyint.
IV. float(n), numeric(*p*,*d*).
V. *null*.
    - char/varchar: caracteres.
    - int/integer, smallint, $\cdots$: valores numéricos.
VI. *Esquema de tablas:*

```
create table departamento
(nombre_depto varchar(20),
edificio varchar(25),
presupuesto numeric(12,2),
primary key(nombre_depto));
```
(1)

Tipos de Datos en SQL

I. char(n) (*n* fijo).
II. varchar(n) (variable -máx *n*-)
III. int/integer, smallint, bigint, tinyint.
IV. float(n), numeric(*p*,*d*).
V. *null*.
   - char/varchar: caracteres.
   - int/integer, smallint, $\cdots$: valores numéricos.
VI. *Esquema de tablas:*

```
create table departamento
(nombre_depto varchar(20),
edificio varchar(25),
presupuesto numeric(12,2),
primary key(nombre_depto));                    (1)
```

Tipos de Datos en SQL

I. char(n) (*n* fijo).
II. varchar(n) (variable -máx *n*-)
III. int/integer, smallint, bigint, tinyint.
IV. float(n), numeric(*p*,*d*).
V. *null*.
   - char/varchar: caracteres.
   - int/integer, smallint, $\cdots$: valores numéricos.
VI. *Esquema de tablas:*

```
create table departamento
(nombre_depto varchar(20),
edificio varchar(25),
presupuesto numeric(12,2),
primary key(nombre_depto));                    (1)
```

Tipos de Datos en SQL

I. char(n) (*n* fijo).
II. varchar(n) (variable -máx *n*-)
III. int/integer, smallint, bigint, tinyint.
IV. float(n), numeric(*p*,*d*).
V. *null*.
   - char/varchar: caracteres.
   - int/integer, smallint, $\cdots$: valores numéricos.
VI. *Esquema de tablas:*

```
create table departamento
(nombre_depto varchar(20),
edificio varchar(25),
presupuesto numeric(12,2),
primary key(nombre_depto));
```
(1)

Tipos de Datos en SQL

I. char(n) (*n* fijo).
II. varchar(n) (variable -máx *n*-)
III. int/integer, smallint, bigint, tinyint.
IV. float(n), numeric(*p*,*d*).
V. *null*.
   - char/varchar: caracteres.
   - int/integer, smallint, $\cdots$: valores numéricos.
VI. *Esquema de tablas:*

create table *departamento*
(*nombre_depto* **varchar**(20),
edificio **varchar**(25),
presupuesto **numeric**(12,2),
primary key(*nombre_depto*)); (1)

Forma general de una Relación/Tabla en SQL

1. *Esquema de tablas (genérico):*

$$\begin{aligned}
&\textbf{create}\ \ \textbf{table}\ r\\
&\quad (A_1\ \ D_1\ ,\\
&\quad A_2\ \ D_2\ ,\\
&\quad \cdots\ ,\\
&\quad A_n\ \ D_n\ ,\\
&\quad \langle \text{integrity-constraint}_1 \rangle,\\
&\quad \cdots\ ,\\
&\quad \langle \text{integrity-constraint}_k \rangle); \qquad (2)
\end{aligned}$$

2. A_I, D_I, $I = 1, 2, \cdots, n$: atributos y dominios.

Forma general de una Relación/Tabla en SQL

- *Esquema de tablas (genérico):*

$$\begin{aligned}
&\textbf{create}\ \ \textbf{table}\ r \\
&\quad (A_1 \quad D_1\ , \\
&\quad A_2 \quad D_2\ , \\
&\quad \cdots\ , \\
&\quad A_n \quad D_n\ , \\
&\quad \langle \text{integrity-constraint}_1 \rangle, \\
&\quad \cdots\ , \\
&\quad \langle \text{integrity-constraint}_k \rangle); \qquad (2)
\end{aligned}$$

- A_I, D_I, $I = 1, 2, \cdots, n$: atributos y dominios.

Integrity Constraints-Ligaduras de Integridad

1. **primary key**$(A_{j_1}, A_{j_2}, \cdots, A_{j_m})$: *no nula (notnull)* y *única (unique)*.
2. **foreign key**$(A_{k_1}, A_{k_2}, \cdots, A_{k_n})$: atributos de relación k en otra relación s.
3. Relaciones con integrity constraints:

create table *departamento*
(*nombre_depto* **varchar**(20),
edificio **varchar**(25),
presupuesto **numeric**(12,2),
primary key(*nombre_depto*)); (3)

create table *cursos*
(*ID_curso* **varchar**(7),
nombre_curso **varchar**(50),
nombre_depto **varchar**(20),
creditos **numeric**(2,0),
primary key(*id_curso*),
foreign key(*nombre_depto*) **references** *departamento*); (4)

Integrity Constraints-Ligaduras de Integridad

1. **primary key**$(A_{j_1}, A_{j_2}, \cdots, A_{j_m})$: *no nula (notnull)* y *única (unique)*.
2. **foreign key**$(A_{k_1}, A_{k_2}, \cdots, A_{k_n})$: atributos de relación k en otra relación s.
3. Relaciones con integrity constraints:

create table *departamento*
(*nombre_depto* **varchar**(20),
edificio **varchar**(25),
presupuesto **numeric**(12,2),
primary key(*nombre_depto*)); (3)

create table *cursos*
(*ID_curso* **varchar**(7),
nombre_curso **varchar**(50),
nombre_depto **varchar**(20),
creditos **numeric**(2,0),
primary key(*id_curso*),
foreign key(*nombre_depto*) **references** *departamento*); (4)

Integrity Constraints-Ligaduras de Integridad

1. **primary key**$(A_{j_1}, A_{j_2}, \cdots, A_{j_m})$: *no nula (notnull)* y *única (unique)*.
2. **foreign key**$(A_{k_1}, A_{k_2}, \cdots, A_{k_n})$: atributos de relación k en otra relación s.
3. Relaciones con integrity constraints:

create table *departamento*
(*nombre_depto* **varchar**(20),
edificio **varchar**(25),
presupuesto **numeric**(12,2),
primary key(*nombre_depto*)); (3)

create table *cursos*
(*ID_curso* **varchar**(7),
nombre_curso **varchar**(50),
nombre_depto **varchar**(20),
creditos **numeric**(2,0),
primary key(*id_curso*),
foreign key(*nombre_depto*) **references** *departamento*); (4)

Modificación de Relaciones

I. **drop table r;** (TODA la relación).
II. **delete from r;** (tuplas y esquema).
    - *r* NO SE BORRA.
III. **alter table r add A D;**
IV. **alter table r drop A;**

Modificación de Relaciones

I. **drop table r;** (TODA la relación).
II. **delete from r;** (tuplas y esquema).
    - *r* NO SE BORRA.
III. **alter table r add A D;**
IV. **alter table r drop A;**

Modificación de Relaciones

I. **drop table r;** (TODA la relación).
II. **delete from r;** (tuplas y esquema).
    - *r* NO SE BORRA.
III. **alter table r add A D;**
IV. **alter table r drop A;**

Modificación de Relaciones

I. **drop table r;** (TODA la relación).
II. **delete from r;** (tuplas y esquema).
    - *r* NO SE BORRA.
III. **alter table r add A D;**
IV. **alter table r drop A;**

Select

1. profesores(ID, nombre, nombre_departamento, salario)
   - Clave principal: ID. Clave foránea: nombre_departamento (relación departamento).
2. Selección de nombre:

```
select nombre
from profesores;
```
(5)

3. Resultado:

Nombre
Wu
Mozart
Einstein
El Said

Select

1. profesores(ID, nombre, nombre_departamento, salario)
   - Clave principal: ID. Clave foránea: nombre_departamento (relación departamento).
2. Selección de nombre:

```
select nombre
from profesores;
```
(5)

3. Resultado:

Nombre
Wu
Mozart
Einstein
El Said

Select

I. profesores(ID, nombre, nombre_departamento, salario)
   - Clave principal: ID. Clave foránea: nombre_departamento (relación departamento).

II. Selección de nombre:

```
select nombre
from profesores;
```
(5)

III. Resultado:

Nombre
Wu
Mozart
Einstein
El Said

Select

I. profesores(ID, nombre, nombre_departamento, salario)
   - nombre_departamento: valores repetidos.

II. Selección de departamento (repetido):

```
select nombre_departamento
from profesores;
```
(6)

III. Selección de departamento (valores **únicos**):

```
select distinct nombre_departamento
from profesores;
```
(7)

IV. Resultado valores **únicos**:

Nombre_departamento
Finanzas
Música
Física
Historia

Select

I. profesores(ID, nombre, nombre_departamento, salario)
   - nombre_departamento: valores repetidos.

II. Selección de departamento (repetido):

```
select nombre_departamento
from profesores;
```
(6)

III. Selección de departamento (valores **únicos**):

```
select distinct nombre_departamento
from profesores;
```
(7)

IV. Resultado valores **únicos**:

Nombre_departamento
Finanzas
Música
Física
Historia

Select

I. profesores(ID, nombre, nombre_departamento, salario)
   - nombre_departamento: valores repetidos.

II. Selección de departamento (repetido):

```
select nombre_departamento
from profesores;
```
(6)

III. Selección de departamento (valores **únicos**):

```
select distinct nombre_departamento
from profesores;
```
(7)

IV. Resultado valores **únicos**:

Nombre_departamento
Finanzas
Música
Física
Historia

Select

I. profesores(ID, nombre, nombre_departamento, salario)
   - nombre_departamento: valores repetidos.

II. Selección de departamento (repetido):

```
select nombre_departamento
from profesores;
```
(6)

III. Selección de departamento (valores **únicos**):

```
select distinct nombre_departamento
from profesores;
```
(7)

IV. Resultado valores **únicos**:

Nombre_departamento
Finanzas
Música
Física
Historia

Select

I. profesores(ID, nombre, nombre_departamento, salario)
   - nombre_departamento: valores repetidos.

II. Selección de departamento (repetido):

select *nombre_departamento*
from *profesores*; (6)

III. Selección de departamento (valores **únicos**):

select distinct *nombre_departamento*
from *profesores*; (7)

IV. Resultado valores **únicos**:

Nombre_departamento
Finanzas
Música
Física
Historia

Select y Condiciones en Atributos

1. Operaciones en valores numéricos:

select *ID*, *nombre_departamento*, salario*1.1
from *profesores*; (8)

2. Condiciones en atributos:

select *nombre*
from *profesores*
where *nombre_departamento*='Historia' **and** *salario* $>$ 8; (9)

- Estructura **where:** where P_1 and/or P_2 and/or $P_3 \cdots$

3. Comparaciones/valores en atributos numéricos:

<, <=, >, >=, =, <>, **is null**, **is unknown** (10)

<> $\Leftrightarrow$! = (NO ES IGUAL).

Select y Condiciones en Atributos

I. Operaciones en valores numéricos:

select *ID*, *nombre_departamento*, salario*1.1
from *profesores*; (8)

II. Condiciones en atributos:

select *nombre*
from *profesores*
where *nombre_departamento*='Historia' **and** *salario* $>$ 8; (9)

- Estructura **where:** where P_1 and/or P_2 and/or $P_3 \cdots$

III. Comparaciones/valores en atributos numéricos:

$<$, $<=$, $>$, $>=$, $=$, $<>$, **is null**, **is unknown** (10)

$<> \Leftrightarrow ! =$ (NO ES IGUAL).

Select y Condiciones en Atributos

I. Operaciones en valores numéricos:

select *ID*, *nombre_departamento*, salario*1.1
from *profesores*; (8)

II. Condiciones en atributos:

select *nombre*
from *profesores*
where *nombre_departamento*='Historia' **and** *salario* $>$ 8; (9)

- Estructura **where:** where P_1 and/or P_2 and/or $P_3 \cdots$

III. Comparaciones/valores en atributos numéricos:

$<$, $<=$, $>$, $>=$, $=$, $<>$, **is null**, **is unknown** (10)

$<> \Leftrightarrow ! =$ (NO ES IGUAL).

Select y Datos Tipo **Char**

I. Inicia con caracter 'a':

$$\textbf{where } A \text{ like } 'a\,\%' \tag{11}$$

II. Termina con caracter 'a':

$$\textbf{where } A \text{ like } '\%a' \tag{12}$$

III. Tiene caracteres 'abc':

$$\textbf{where } A \text{ like } '\%abc\,\%' \tag{13}$$

IV. Empieza con 'a' y termina con 'b':

$$\textbf{where } A \text{ like } 'a\,\%b' \tag{14}$$

V. El segundo caracter es 'a':

$$\textbf{where } A \text{ like } '\_a\,\%' \tag{15}$$

Select y Datos Tipo **Char**

I. Inicia con caracter 'a':

$$\textbf{where } A \text{ like } 'a\,\%' \tag{11}$$

II. Termina con caracter 'a':

$$\textbf{where } A \text{ like } '\,\%a' \tag{12}$$

III. Tiene caracteres 'abc':

$$\textbf{where } A \text{ like } '\,\%abc\,\%' \tag{13}$$

IV. Empieza con 'a' y termina con 'b':

$$\textbf{where } A \text{ like } 'a\,\%b' \tag{14}$$

V. El segundo caracter es 'a':

$$\textbf{where } A \text{ like } '\_a\,\%' \tag{15}$$

Select y Datos Tipo **Char**

I. Inicia con caracter 'a':

$$\textbf{where } A \text{ like } 'a\,\%' \tag{11}$$

II. Termina con caracter 'a':

$$\textbf{where } A \text{ like } '\,\%a' \tag{12}$$

III. Tiene caracteres 'abc':

$$\textbf{where } A \text{ like } '\,\%abc\,\%' \tag{13}$$

IV. Empieza con 'a' y termina con 'b':

$$\textbf{where } A \text{ like } 'a\,\%b' \tag{14}$$

V. El segundo caracter es 'a':

$$\textbf{where } A \text{ like } '\_a\,\%' \tag{15}$$

Select y Datos Tipo **Char**

I. Inicia con caracter 'a':

$$\textbf{where } A \text{ like } 'a\,\%' \tag{11}$$

II. Termina con caracter 'a':

$$\textbf{where } A \text{ like } '\,\%a' \tag{12}$$

III. Tiene caracteres 'abc':

$$\textbf{where } A \text{ like } '\,\%abc\,\%' \tag{13}$$

IV. Empieza con 'a' y termina con 'b':

$$\textbf{where } A \text{ like } 'a\,\%b' \tag{14}$$

V. El segundo caracter es 'a':

$$\textbf{where } A \text{ like } '\_a\,\%' \tag{15}$$

Select y Datos Tipo **Char**

I. Inicia con caracter 'a':

$$\textbf{where } A \text{ like } 'a\,\%' \tag{11}$$

II. Termina con caracter 'a':

$$\textbf{where } A \text{ like } '\,\%a' \tag{12}$$

III. Tiene caracteres 'abc':

$$\textbf{where } A \text{ like } '\,\%abc\,\%' \tag{13}$$

IV. Empieza con 'a' y termina con 'b':

$$\textbf{where } A \text{ like } 'a\,\%b' \tag{14}$$

V. El segundo caracter es 'a':

$$\textbf{where } A \text{ like } '\_a\,\%' \tag{15}$$

Select y Datos Tipo **Char**

1. 3 caracteres exactamente:

$$\textbf{where } A \text{ like } '\_\ \_\ \_' \tag{16}$$

2. Al menos con 3 caracteres:

$$\textbf{where } A \text{ like } '\_\ \_\ \_\ \%' \tag{17}$$

3. *Expresiones regulares -regex-*:

$$\begin{aligned} '\hat{}c\,\%' &: \text{comienza con c} \\ '(a|o)\,\%' &: \text{comienza con c y sigue a/o} \\ '.*\,\%' &: \text{cualquier string} \end{aligned} \tag{18}$$

Select y Datos Tipo **Char**

I. 3 caracteres exactamente:

$$\textbf{where } A \text{ like } '\_\ \_\ \_' \tag{16}$$

II. Al menos con 3 caracteres:

$$\textbf{where } A \text{ like } '\_\ \_\ \_\ \%' \tag{17}$$

III. *Expresiones regulares -regex-:*

$$\begin{aligned} '\hat{}c\ \%' &: \text{comienza con c} \\ '(a|o)\ \%' &: \text{comienza con c y sigue a/o} \\ '.*\ \%' &: \text{cualquier string} \end{aligned} \tag{18}$$

Select y Datos Tipo **Char**

I. 3 caracteres exactamente:

$$\textbf{where } A \text{ like } '\_\ \_\ \_' \tag{16}$$

II. Al menos con 3 caracteres:

$$\textbf{where } A \text{ like } '\_\ \_\ \_\ \%' \tag{17}$$

III. *Expresiones regulares -regex-*:

$$\begin{aligned} '\hat{}c\,\%' &: && \text{comienza con c} \\ '(a|o)\,\%' &: && \text{comienza con c y sigue a/o} \\ '.*\ \%' &: && \text{cualquier string} \end{aligned} \tag{18}$$

Select con Varias Relaciones

1. Estructura general:

$$\begin{aligned} &\textbf{select } A_1, A_2, \cdots, A_n \\ &\textbf{from } r_1, r_2, \cdots, r_m \\ &\text{where P;} \end{aligned} \tag{19}$$

2. Ejemplo: dos relaciones tipo

   profesores(ID, nombre, nombre_departamento, salario)

   ensenya(ID, ID_curso, ID_seccion, semestre, anyo)

   Operación de selección:

   select *nombre, ID_curso* **from** *profesores, ensenya*

   where *profesor.ID = ensenya.ID*

3. Resultado:

nombre	ID_curso
Wu	FIN-201
...	...

Select con Varias Relaciones

- Estructura general:

$$\begin{aligned} &\textbf{select } A_1, A_2, \cdots, A_n \\ &\textbf{from } r_1, r_2, \cdots, r_m \\ &\text{where P;} \end{aligned} \tag{19}$$

- Ejemplo: dos relaciones tipo

  profesores(ID, nombre, nombre_departamento, salario)

  ensenya(ID, ID_curso, ID_seccion, semestre, anyo)

  Operación de selección:

  select *nombre, ID_curso* **from** *profesores, ensenya*

  where *profesor.ID* = *ensenya.ID*

- Resultado:

nombre	ID_curso
Wu	FIN-201
...	...

Select con Varias Relaciones

I. Estructura general:

$$\begin{aligned} &\textbf{select } A_1, A_2, \cdots, A_n \\ &\textbf{from } r_1, r_2, \cdots, r_m \\ &\text{where P;} \end{aligned} \tag{19}$$

II. Ejemplo: dos relaciones tipo

profesores(ID, nombre, nombre_departamento, salario)

ensenya(ID, ID_curso, ID_seccion, semestre, anyo)

Operación de selección:

select *nombre, ID_curso* **from** *profesores, ensenya*

where *profesor.ID* = *ensenya.ID*

III. Resultado:

nombre	ID_curso
Wu	FIN-201
...	...

Otras propiedades de Select

I. Álgebra relacional de una selección en múltiples relaciones:

$$\Pi_{A_1, A_2, \cdots, A_n}(\sigma_P(r_1 \times r_2 \times \cdots \times r_m)) \tag{20}$$

II. Selección de todos los atributos:

select * **from** *r* **where** P; (21)

III. Ordenar/agrupar la selección por atributos:

select * **from** *r* **order by** *A* **desc/asc**;

select * **from** *r* **group by** *A*; (22)

IV. Seleccionar con intervalos:

select * **from** *r* **where** *A* **between** valor1 **and** valor2; (23)

V. Renombrar tablas:

select t1.A_1, t2.A_2
from *tabla 1* **as** t1, *tabla2* **as** t2
where P; (24)

Otras propiedades de Select

I. Álgebra relacional de una selección en múltiples relaciones:

$$\Pi_{A_1, A_2, \cdots, A_n}(\sigma_P(r_1 \times r_2 \times \cdots \times r_m)) \tag{20}$$

II. Selección de todos los atributos:

select * **from** *r* **where** P; (21)

III. Ordenar/agrupar la selección por atributos:

select * **from** *r* **order by** *A* **desc/asc**;

select * **from** *r* **group by** *A*; (22)

IV. Seleccionar con intervalos:

select * **from** *r* **where** *A* **between** valor1 **and** valor2; (23)

V. Renombrar tablas:

select t1.A_1, t2.A_2
from *tabla 1* **as** t1, *tabla2* **as** t2
where P; (24)

Otras propiedades de Select

I. Álgebra relacional de una selección en múltiples relaciones:

$$\Pi_{A_1, A_2, \cdots, A_n}(\sigma_P(r_1 \times r_2 \times \cdots \times r_m)) \tag{20}$$

II. Selección de todos los atributos:

select * **from** *r* **where** P; (21)

III. Ordenar/agrupar la selección por atributos:

select * **from** *r* **order by** *A* **desc/asc**;

select * **from** *r* **group by** *A*; (22)

IV. Seleccionar con intervalos:

select * **from** *r* **where** *A* **between** valor1 **and** valor2; (23)

V. Renombrar tablas:

select t1.A_1, t2.A_2
from *tabla 1* **as** t1, *tabla2* **as** t2
where P; (24)

Otras propiedades de Select

I. Álgebra relacional de una selección en múltiples relaciones:

$$\Pi_{A_1, A_2, \cdots, A_n}(\sigma_P(r_1 \times r_2 \times \cdots \times r_m)) \tag{20}$$

II. Selección de todos los atributos:

select * **from** *r* **where** P; (21)

III. Ordenar/agrupar la selección por atributos:

select * **from** *r* **order by** *A* **desc/asc**;

select * **from** *r* **group by** *A*; (22)

IV. Seleccionar con intervalos:

select * **from** *r* **where** *A* **between** valor1 **and** valor2; (23)

V. Renombrar tablas:

select t1.A_1, t2.A_2
from *tabla 1* **as** t1, *tabla2* **as** t2
where P; (24)

Otras propiedades de Select

I. Álgebra relacional de una selección en múltiples relaciones:

$$\Pi_{A_1, A_2, \cdots, A_n}(\sigma_P(r_1 \times r_2 \times \cdots \times r_m)) \tag{20}$$

II. Selección de todos los atributos:

select * **from** *r* **where** P; (21)

III. Ordenar/agrupar la selección por atributos:

select * **from** *r* **order by** *A* **desc/asc**;

select * **from** *r* **group by** *A*; (22)

IV. Seleccionar con intervalos:

select * **from** *r* **where** *A* **between** valor1 **and** valor2; (23)

V. Renombrar tablas:

select t1.A_1, t2.A_2

from *tabla 1* **as** t1, *tabla2* **as** t2

where P; (24)

Otras Operaciones Binarias con Select

- Unión:

(select A_1 **from** r_1 **where** P1)
union/union all
(select A_2 **from** r_1 **where** P2); (25)

- Intersección:

(select A_1 **from** r_1 **where** P1)
intersect/intersect all
(select A_2 **from** r_1 **where** P2); (26)

- Excepción:

(select A_1 **from** r_1 **where** P1)
except
(select A_2 **from** r_1 **where** P2); (27)

union/intersect/except all: retiene duplicados.

Otras Operaciones Binarias con Select

I. Unión:

(**select** A_1 **from** r_1 **where** P1)
union/union all
(**select** A_2 **from** r_1 **where** P2); (25)

II. Intersección:

(**select** A_1 **from** r_1 **where** P1)
intersect/intersect all
(**select** A_2 **from** r_1 **where** P2); (26)

III. Excepción:

(**select** A_1 **from** r_1 **where** P1)
except
(**select** A_2 **from** r_1 **where** P2); (27)

union/intersect/except all: retiene duplicados.

Otras Operaciones Binarias con Select

I. Unión:

(**select** A_1 **from** r_1 **where** P1)
union/union all
(**select** A_2 **from** r_1 **where** P2); (25)

II. Intersección:

(**select** A_1 **from** r_1 **where** P1)
intersect/intersect all
(**select** A_2 **from** r_1 **where** P2); (26)

III. Excepción:

(**select** A_1 **from** r_1 **where** P1)
except
(**select** A_2 **from** r_1 **where** P2); (27)

union/intersect/except all: retiene duplicados.

Funciones Sobre Atributos

1. Listado:

Función	Sintaxis
Promedio	**avg**
Mínimo/Máximo	**min/max**
Suma total	**sum**
Conteo	**count**

- Ejemplo 1: salario promedio de profesores en un departamento:

select **avg**(*salario*) **as** *sal_prom*

from *profesores*

where *nombre_departamento* = 'Historia'; (28)

- Ejemplo 2: salario promedio de profesores en un departamento agrupados por departamento:

select *nombre_departamento*, **avg**(*salario*) **as** *sal_prom*

from *profesores*

group by *nombre_departamento*; (29)

- *Nota:* **having** condiciona la selección de la función

Funciones Sobre Atributos

1. Listado:

Función	Sintaxis
Promedio	**avg**
Mínimo/Máximo	**min/max**
Suma total	**sum**
Conteo	**count**

- Ejemplo 1: salario promedio de profesores en un departamento:

select **avg**(*salario*) **as** *sal_prom*

from *profesores*

where *nombre_departamento* = 'Historia'; (28)

- Ejemplo 2: salario promedio de profesores en un departamento agrupados por departamento:

select *nombre_departamento*, **avg**(*salario*) **as** *sal_prom*

from *profesores*

group by *nombre_departamento*; (29)

- *Nota:* **having** condiciona la selección de la función

Funciones Sobre Atributos

1. Listado:

Función	Sintaxis
Promedio	**avg**
Mínimo/Máximo	**min/max**
Suma total	**sum**
Conteo	**count**

- Ejemplo 1: salario promedio de profesores en un departamento:

select **avg**(*salario*) **as** *sal_prom*

from *profesores*

where *nombre_departamento* = 'Historia'; (28)

- Ejemplo 2: salario promedio de profesores en un departamento agrupados por departamento:

select *nombre_departamento*, **avg**(*salario*) **as** *sal_prom*

from *profesores*

group by *nombre_departamento*; (29)

- *Nota:* **having** condiciona la selección de la función

Funciones Sobre Atributos

1. Listado:

Función	Sintaxis
Promedio	**avg**
Mínimo/Máximo	**min/max**
Suma total	**sum**
Conteo	**count**

- Ejemplo 1: salario promedio de profesores en un departamento:

select **avg**(*salario*) **as** *sal_prom*

from *profesores*

where *nombre_departamento* = 'Historia'; (28)

- Ejemplo 2: salario promedio de profesores en un departamento agrupados por departamento:

select *nombre_departamento*, **avg**(*salario*) **as** *sal_prom*

from *profesores*

group by *nombre_departamento*; (29)

- *Nota:* **having** condiciona la selección de la función.

Anidamiento/*Nesting*

- "Intersección":

  select distinct *course_id* **from** *section*
  where *semester* = 'Fall' **and** *year* = 2017 **and** *course_id* **in**
  (**select** *course_id* **from** *section*
  where *semester* = 'Spring' **and** *year* = 2018); (30)

- En lugar de **in**;
  - **not in**
  - **all**

    where *salary* > **all** (**select** *salary* **from** instructor
    where *dept_name* = 'Biology')

  - **exists:** tuplas no vacías.
  - **unique:** tuplas no duplicadas.

Anidamiento/*Nesting*

I. "Intersección":

select distinct *course_id* **from** *section*
where *semester* = 'Fall' **and** *year* = 2017 **and** *course_id* **in**
(**select** *course_id* **from** *section*
where *semester* = 'Spring' **and** *year* = 2018); (30)

II. En lugar de **in**;

- **not in**
- **all**

where *salary* > **all** (**select** *salary* **from** instructor
where *dept_name* = 'Biology')

- **exists:** tuplas no vacías.
- **unique:** tuplas no duplicadas.

Anidamiento/*Nesting*

I. "Intersección":

select distinct *course_id* **from** *section*
where *semester* = 'Fall' **and** *year* = 2017 **and** *course_id* **in**
(**select** *course_id* **from** *section*
where *semester* = 'Spring' **and** *year* = 2018); (30)

II. En lugar de **in**;
- **not in**
- **all**

where *salary* > **all** (**select** *salary* **from** instructor
where *dept_name* = 'Biology')

- **exists:** tuplas no vacías.
- **unique:** tuplas no duplicadas.

Anidamiento/*Nesting*

I. "Intersección":

select distinct *course_id* **from** *section*
where *semester* = 'Fall' **and** *year* = 2017 **and** *course_id* **in**
(**select** *course_id* **from** *section*
where *semester* = 'Spring' **and** *year* = 2018); (30)

II. En lugar de **in**;
- **not in**
- **all**

where *salary* > **all** (**select** *salary* **from** instructor
where *dept_name* = 'Biology')

- **exists:** tuplas no vacías.
- **unique:** tuplas no duplicadas.

Anidamiento/*Nesting*

I. "Intersección":

select distinct *course_id* **from** *section*
where *semester* = 'Fall' **and** *year* = 2017 **and** *course_id* **in**
(**select** *course_id* **from** *section*
where *semester* = 'Spring' **and** *year* = 2018); (30)

II. En lugar de **in**;

- **not in**
- **all**

where *salary* > **all** (**select** *salary* **from** instructor
where *dept_name* = 'Biology')

- **exists:** tuplas no vacías.
- **unique:** tuplas no duplicadas.

Anidamiento/*Nesting*

I. "Intersección":

select distinct *course_id* **from** *section*
where *semester* = 'Fall' **and** *year* = 2017 **and** *course_id* **in**
(**select** *course_id* **from** *section*
where *semester* = 'Spring' **and** *year* = 2018); (30)

II. En lugar de **in**;

- **not in**
- **all**

where *salary* > **all** (**select** *salary* **from** instructor
where *dept_name* = 'Biology')

- **exists:** tuplas no vacías.
- **unique:** tuplas no duplicadas.

Anidamiento/*Nesting*

- Relaciones temporales con **with**:

with *max_budget*(*value*) **as**
(**select max**(*budget*)
from *department*)
select *budget*
from *department*, *max_budget*
where *department.budget* = *max_budget.value*; (31)

  - Relación temporal: *max_budget*. Atributo: *value*.
  - *value*: **select max**(*budget*) **from** *department*.

Anidamiento/*Nesting*

- Relaciones temporales con **with**:

with *max_budget*(*value*) **as**
(**select max**(*budget*)
from *department*)
select *budget*
from *department*, *max_budget*
where *department.budget* = *max_budget.value*; (31)

  - Relación temporal: *max_budget*. Atributo: *value*.
  - *value*: **select max**(*budget*) **from** *department*.

Anidamiento/*Nesting*

- Relaciones temporales con **with**:

with *max_budget*(*value*) **as**
(**select max**(*budget*)
from *department*)
select *budget*
from *department*, *max_budget*
where *department.budget* = *max_budget.value*; (31)

  - Relación temporal: *max_budget*. Atributo: *value*.
  - *value*: **select max**(*budget*) **from** *department*.

Ejemplo con **with**

- Relación *employee*:

ID	Name	Salary
100011	Smith	50000
100022	Bill	94000
100027	Sam	70550
100845	Walden	80000
115585	Erik	60000
117007	Kate	69000

  - Valor del salario promedio:

$$\textbf{avg}(\textit{salary}) = 70591 \tag{32}$$

  - Lista empleados con salario mayor al promedio = ¿?

Ejemplo con **with**

- Relación *employee*:

ID	Name	Salary
100011	Smith	50000
100022	Bill	94000
100027	Sam	70550
100845	Walden	80000
115585	Erik	60000
117007	Kate	69000

  - Valor del salario promedio:

$$\textbf{avg}(\textit{salary}) = 70591 \tag{32}$$

  - Lista empleados con salario mayor al promedio = ¿?

Ejemplo con **with**

- Relación *employee*:

ID	Name	Salary
100011	Smith	50000
100022	Bill	94000
100027	Sam	70550
100845	Walden	80000
115585	Erik	60000
117007	Kate	69000

  - Valor del salario promedio:

$$\textbf{avg}(\textit{salary}) = 70591 \tag{32}$$

  - Lista empleados con salario mayor al promedio = ¿?

Ejemplo con **with**

1. Petición:

$$\begin{aligned}&\textbf{with } \textit{temptable}(\textit{avgval}) \textbf{ as}\\ &\quad (\textbf{select avg}(\textit{Salary})\\ &\quad \textbf{from } \textit{Employee})\\ &\quad \textbf{select } \textit{ID}, \textit{Name}, \textit{Salary}\\ &\quad \textbf{from } \textit{Employee}, \textit{temptable}\\ &\quad \textbf{where } \textit{Employee.Salary} > \textit{temptable.avgval};\end{aligned} \tag{33}$$

2. Resultado:

ID	Name	Salary
100022	Bill	94000
100845	Walden	80000

Ejemplo con **with**

I. Petición:

with *temptable*(*avgval*) **as**
(**select avg**(*Salary*)
from *Employee*)
select *ID*, *Name*, *Salary*
from *Employee*, *temptable*
where *Employee.Salary* > *temptable.avgval*; (33)

II. Resultado:

ID	Name	Salary
100022	Bill	94000
100845	Walden	80000

Operaciones en las Relaciones

I. Borrar:

delete from *instructor* **where** *dept_name* = 'Finance';

delete from *instructor* **where** *salary* **between** 13000 **and** 15000; (34)

II. Insertar:

insert into *course*(*course_id, title, dept_name, credits*)
values('CS-437', 'Database Systems',
'Comp. Sci', 4); (35)

III. Actualizar:

update instructor
set *salary* = *salary**1.05; (36)

Operaciones en las Relaciones

I. Borrar:

delete from *instructor* **where** *dept_name* = 'Finance';

delete from *instructor* **where** *salary* **between** 13000 **and** 15000; (34)

II. Insertar:

insert into *course*(*course_id*, *title*, *dept_name*, *credits*)
values('CS-437', 'Database Systems',
'Comp. Sci', 4); (35)

III. Actualizar:

update instructor
set *salary* = *salary**1.05; (36)

Operaciones en las Relaciones

I. Borrar:

delete from *instructor* **where** *dept_name* = 'Finance';

delete from *instructor* **where** *salary* **between** 13000 **and** 15000; (34)

II. Insertar:

insert into *course*(*course_id, title, dept_name, credits*)
values('CS-437', 'Database Systems',
'Comp. Sci', 4); (35)

III. Actualizar:

update instructor
set *salary* = *salary**1.05; (36)

Operaciones en las Relaciones

1. Dos relaciones en una DB universitaria:

   estudiante(ID, nombre, nombre_departamento, creditos)

   cursosinsc(ID, ID_curso, ID_seccion, semestre, anyo, nota)) (37)

ID	nombre	nombre_departamento	creditos
...	...	...	...

ID	ID_curso	ID_seccion	semestre	anyo	nota
...	...	...	...	...	...

2. Query para Join:

   select *nombre, nombre_departamento, ID_curso*

   from *estudiante, cursosinsc*

   where *estudiante.ID = cursosinsc.ID*; (38)

3. Resultado:

   join(nombre, nombre_departamento, ID_curso) (39)

Operaciones en las Relaciones

1. Dos relaciones en una DB universitaria:

   estudiante(ID, nombre, nombre_departamento, creditos)

   cursosinsc(ID, ID_curso, ID_seccion, semestre, anyo, nota)) (37)

ID	nombre	nombre_departamento	creditos
...	...	...	...

ID	ID_curso	ID_seccion	semestre	anyo	nota
...	...	...	...	...	...

2. Query para Join:

   select *nombre, nombre_departamento, ID_curso*

   from *estudiante, cursosinsc*

   where *estudiante.ID* = *cursosinsc.ID*; (38)

3. Resultado:

   join(nombre, nombre_departamento, ID_curso) (39)

Operaciones en las Relaciones

I. Dos relaciones en una DB universitaria:

estudiante(ID, nombre, nombre_departamento, creditos)

cursosinsc(ID, ID_curso, ID_seccion, semestre, anyo, nota)) (37)

ID	nombre	nombre_departamento	creditos
...	...	...	...

ID	ID_curso	ID_seccion	semestre	anyo	nota
...	...	...	...	...	...

II. Query para Join:

select *nombre, nombre_departamento, ID_curso*

from *estudiante, cursosinsc*

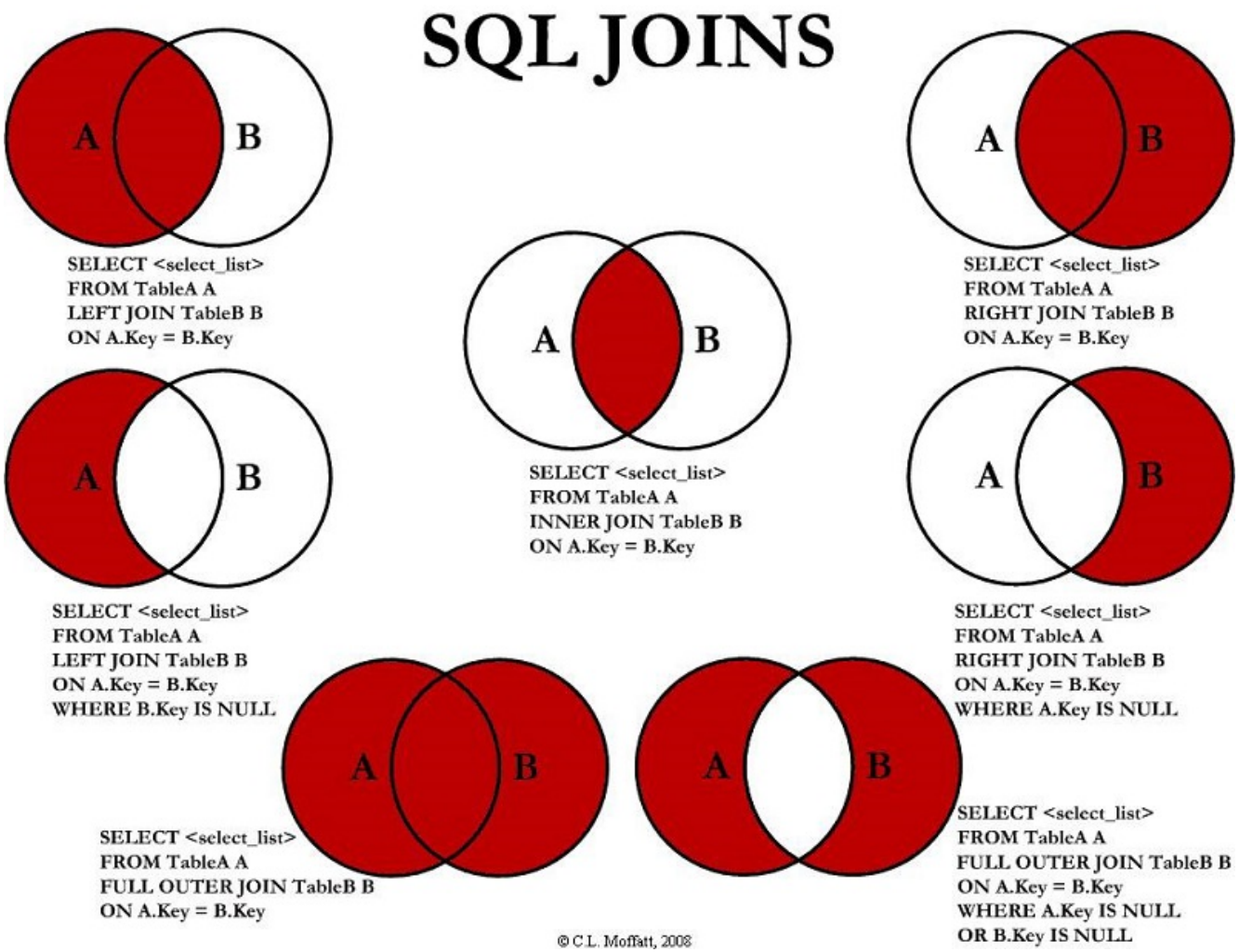
where *estudiante.ID* = *cursosinsc.ID*; (38)

III. Resultado:

join(nombre, nombre_departamento, ID_curso) (39)

Operaciones en las Relaciones

1. Tipos de Join:



Ejemplo de View

1. Sintaxis:

create view *facultad* **as**
select *ID*, *nombre*, *nombre_departamento*
from *profesores*; (40)

Consulta:

select * from *facultad*;

2. Sintaxis más compleja:

create view *cursos_fisica* **as**
select *ID*, *ID_curso*, *seccion*
from *profesores*, *ensenya*
where *profesores.ID* = *ensenya.ID*
and *profesores.nombre_departamento* = *'Física'*; (41)

Ejemplo de View

I. Sintaxis:

create view *facultad* **as**
select *ID, nombre, nombre_departamento*
from *profesores*; (40)

Consulta:

select * from *facultad*;

II. Sintaxis más compleja:

create view *cursos_fisica* **as**
select *ID, ID_curso, seccion*
from *profesores, ensenya*
where *profesores.ID = ensenya.ID*
and *profesores.nombre_departamento = 'Física'*; (41)

Ejercicio DB

1. De la página oficial del DANE Microdatos, descargue el archivo correspondiente a enero de 2012.
2. Calcule la cantidad total de países de origen y países de procedencia.
3. Calcule la suma total del valor CIF en Pesos y en Dólares tomando en cuenta la siguiente ligadura: país de origen = país de procedencia.
4. Calcule esta suma agrupando el resultado por países.
5. Encuentre el total de actividades económicas que tiene cada país de origen.
6. Cree una **view** con los siguientes elementos: país procedencia, bandera, PBK, valor en dólares y valor en pesos.